UNIT-3
PUBLIC KEY CRYPTOGRAPHY

MATHEMATICS OF ASYMMETRIC KEY CRYPTOGRAPHY: Primes – Primality Testing – Factorization – Euler's totient function, Fermat's and Euler's Theorem – Chinese Remainder Theorem – Exponentiation and logarithm – ASYMMETRIC KEY CIPHERS: RSA cryptosystem – Key distribution – Key management – Diffie Hellman key exchange -ElGamal cryptosystem – Elliptic curve arithmetic-Elliptic curve cryptography.

PRIME NUMBER

An integer $p > 1$ is a prime number if and only if its only divisors are ±1 and ±p.
Any integer $a > 1$ can be factored in a unique way as

$$a = p_1^{a_1} \times p_2^{a_2} \times \cdots \times p_t^{a_t}$$

where $p_1 < p_2 < \ldots < p_t$ are prime numbers and where each a_i is a positive integer.

$91 = 7 * 13$
$3600 = 2^4 * 3^2 * 5^2$
$11011 = 7 * 11^2 * 13$

If P is the set of all prime numbers, then any positive integer a can be written uniquely in the following form:

$$a = \prod_{p \in P} p^{a_p} \quad \text{where each } a_p \geq 0$$

It is easy to determine the greatest common divisor of two positive integers if we express each integer as the product of primes.

$300 = 2^2 * 3^1 * 5^2$
$18 = 2^1 * 3^2$
$\gcd(18, 300) = 2^1 * 3^1 * 5^0 = 6$

The following relationship always holds:
If $k = \gcd(a, b)$, then $k_p = \min(a_p, b_p)$ for all p.

FERMAT'S THEOREMS

Fermat's theorem states the following: if p is a prime and a is a positive integer not divisible by p, then

$$\mathbf{a^{p-1} \equiv 1 (mod\ p)}$$

Consider the set of positive integers less than p: {1,2,3..p-1}
Multiply each element by *a modulo p* to get the set

X ={ a mod p,2a mod p….(p-1)mod p}.

None of the elements of X is equal to zero because p does not divide a. No two of the integers in X are equal.(p-1) elements of X are all positive integers with no two elements are equal. Multiplying the numbers in both sets and taking the result mod p yields.

$a * 2a *\ldots*(p-1)a \equiv [(1*2*\ldots*(p-1)](mod\ p)$
$\{1 * 2 *\ldots*(p-1)\}\ a^{p-1} \equiv [(1*2*\ldots*(p-1)](mod\ p)$
$(p-1)!\ a^{p-1} \equiv (p-1)!(mod\ p)$

$$\mathbf{a^{p-1} \equiv 1 (mod\ p)}$$

Example
$a = 7, p = 19$
$7^2 = 49 \equiv 11 \pmod{19}$
$7^4 = 121 \equiv 7 \pmod{19}$
$7^8 \equiv 49 \equiv 11 \pmod{19}$
$7^{16} \equiv 121$ K 7 (mod 19)
$a^{p-1} = 7^{18} = 7^{16} * 7^2 \equiv 7 * 11 \equiv 1$ (mod 19

An alternative form of Fermat's theorem is also useful: If *p* is prime and *a* is a positive integer, then

$$\mathbf{a^p \equiv a \pmod{p}}$$

Euler's totient function

It is represented as *ø(n).*Euler's totient function is defined as the number of positive integers less than *n* and relatively prime to *n*. *ø(1)=1*

It should be clear that for a prime number *p*

$$\boldsymbol{\phi(p)=p-1}$$

Suppose that we have two prime numbers *p* and *q*, with *p* not equal to *q*. Then we can show that

$$n=pq.$$
$$\mathbf{\phi(n)= \phi(pq)= \phi(p)* \phi(q)=(p-1)*(q-1)}$$
$$\phi(n)=(pq-1)-[(q-1)+(p-1)]$$
$$= pq-(p+q)+1$$
$$=(p-1)*(q-1)$$
$$= \phi(p)* \phi(q)$$

$$\boldsymbol{\phi(p^n) = p^n\ (1 - \frac{1}{p}}$$

To determine f(35), we list all of the positive integers less than 35 that are relatively prime to it:
1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18
19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34
There are 24 numbers on the list, so f(35) = 24.

f(21) = f(3) * f(7) = (3 - 1) * (7 - 1) = 2 * 6 = 12

EULER'S THEOREM

Euler's theorem states that for every *a* and *n* that are relatively prime:

$$\mathbf{a^{\phi(n)}=1 \pmod{n}}$$

The above equation is true, if n is prime, because in that case *ø(n)=(n-1)* and Fermat's theorem holds. However it holds for any integer *n*. recall that ø(n)is the number of positive integers less than n that are relatively prime to *n*. consider the set of such integers, labeled as follows:
$R=\{x_1, x_2 \ldots x_{\phi(n)}\}$
That is, each element x_i of R is a unique positive integer less than *n* with $\gcd(x_i, n)=1$.now multiply each element by *a modulo n:*
$S=\{(ax_1 \bmod n), (ax_2 \bmod n), \ldots (ax_{\phi(n)}) \bmod n)\}$
The set S is a permutation of R, by the following reasons:
1. Because *a* is relatively prime to *n* and x_i is relatively prime to *n*,ax_i must also be relatively prime to *n*. thus all the members of S are integers that are less than n and that are relatively prime to n.
2. There are no duplicates in S. if $ax_i \bmod n = ax_i \bmod n$, then $x_i = x_j$

$$\prod_{i=1}^{\phi(n)} (ax_i \bmod n) = \prod_{i=1}^{\phi(n)} x_i$$

$$\prod_{i=1}^{\phi(n)} ax_i \equiv \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \times \left[\prod_{i=1}^{\phi(n)} x_i\right] \equiv \prod_{i=1}^{\phi(n)} x_i \pmod{n}$$

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

An alternative form of the theorem is also useful:

$$a^{\phi(n)+1} \equiv a \pmod{n}$$

$$a = 3; n = 10; \phi(10) = 4 \; a^{\phi(n)} = 3^4 = 81 = 1 \pmod{10} = 1 \pmod{n}$$
$$a = 2; n = 11; \phi(11) = 10 \; a^{\phi(n)} = 2^{10} = 1024 = 1 \pmod{11} = 1 \pmod{n}$$

TESTING FOR PRIMALITY

For many cryptographic algorithms, it is necessary to select one or more very large prime numbers at random. Thus, we are faced with the task of determining whether a given large number is prime. There is no simple yet efficient means of accomplishing this task.

Miller-Rabin Algorithm

The algorithm due to Miller and Rabin [MILL75, RABI80] is typically used to test a large number for primality.

TEST (*n*)

1. Find integers *k*, *q*, with $k > 0$, *q* odd, so that $(n - 1 = 2^k q)$;
2. Select a random integer *a*, $1 < a < n - 1$;
3. **if** $a^q \bmod n = 1$ **then** return("inconclusive");
4. **for** $j = 0$ **to** $k - 1$ **do**
5. **if** $a^{2^j q} \bmod n = n - 1$ **then** return("inconclusive");
6. return("composite");

Let us apply the test to the prime number $n = 29$.
$(n - 1) = 28 = 2^2(7) = 2^k q$.
First, let us try $a = 10$.
Compute $10^7 \bmod 29 = 17$,

j=0; $10^{2*0*7} \bmod 29 = 1$
j=1; $10^{2*1*7} \bmod 29 = 10^{14} \bmod 19 = 28$, and the test returns inconclusive.
So n is prime number.

$$10^2 \bmod 29 \equiv 13 \bmod 19$$
$$10^4 \bmod 29 \equiv 10^{2*} 10^2 \bmod 29 = (13 * 13) \bmod 19 = 24 \bmod 29$$
$$10^8 \bmod 29 \equiv 10^{4*} 10^4 \bmod 29 = (24 * 24) \bmod 19 = 25 \bmod 29$$
$$10^{14} \bmod 29 \equiv 10^{8*} 10^4 * 10^2 \bmod 29 = (25 * 24 * 13) \bmod 19 = 7800 \bmod 29 = 28$$

FACTORIZATION

factorization techniques

1. Fermat's **factoring**,
2. Pollards p-1 **method**

Fermat's Factoring

Algorithm:

```
fermat_factorization(n)    //n is the prime number
 {
   x        sqrt(n)
   while(x<n)
     {
        w        x^2 – n
       if(w is perfect square)
         y         sqrt(w);
          a=x+y
           b=x-y
          return a&b;
         x=x+1
}}
```

Find i) 3811 using Fermat's Factoring

w	**Perfect Sqrt(w)**	
62^2 -3811=3844-3811=33	NIL	
63^2 -3811=3969-3811=158	NIL	
64^2 -3811=4096-3811=285	NIL	
65^2 -3811=4225-3811=414	NIL	
66^2 -3811=4356-3811=545	NIL	
67^2 -3811=4489-3811=678	NIL	
68^2 -3811=4624-3811=813	NIL	
69^2 -3811=4761-3811=950	NIL	
70^2 -3811=4900-3811=1089	33^2	$=70^2 - 33^2$ =(70+33)(70-33)=103 * 37 RESULT: 103*37

Pollard-(p-1) Factorization

Alogrithm:

```
Pollard_(p-1)_Factorization(n,B)
 {
   B=n
   a=2
   e=2
 while(e<=B)
  {
   a=a^e  modn
    e=e+1
```

}
P=gcd(a-1,n)
If i<p<n return p
Else
return failure
}

N=1403		
a=2,e=2		
While(2<=1403) $a=2^2$ mod1403 a=4 e=3	gcd(3,1403)=1	
While(3<=1403) $a=4^3$ mod1403 =64mod1403 =64	Gcd(63,1403)=1	
While(4<=1403) $a=64^4$ mod1403 =142	gcd(141,1403)=1	
While(5<=1403) $a=142^5$ mod1403 =794	gcd(793,1403)=61	=1403/61=23 =61*23 Factor=61*23

THE CHINESE REMAINDER THEOREM

The Chinese Remainder Theorem says it is possible to reconstruct integers in certain range from their residues modulo a set of pair wise relatively prime moduli.

$x \equiv a_1 \bmod n_1$, $x \equiv a_2 \bmod n_2$, $x \equiv a_k \bmod n_k$

If $n_1,n_2,..,n_k$ are positive integers that are pairwise co-prime and $a_1,a_2,\ldots,a_k$ are any integers, then CRT is used to find the values of x that solves the following congruence simultaneously.

Value of $x=(a_1m_1y_1+a_2m_2y_2+\ldots+a_km_ky_k) \bmod M$

Where $M=n_1n_2n_3..n_k$

$m_i=M/n_i$

$m_iy_i=1 \bmod n_i$

Problem 1

$x \equiv 1 \bmod 5$
$x \equiv 2 \bmod 6$
$x \equiv 3 \bmod 7$

$a_1=1$
$a_2=2$
$a_3=3$

$n_1=5$
$n_2=6$
$n_3=7$

$M=n_1n_2n_3$
$M=5*6*7=210$

$m_i=M/n_i$

$m1=210/5=42$
$m2=210/6=35$
$m3=210/7=30$

$m_iy_i=1 \bmod n_i$
$42y_1=1 \bmod 5$
$y_1=3 \bmod 5$

$35y_2= 1 \bmod 6$
$y_2=5 \bmod 6$

$30y_3=1 \bmod 7$
$y_3=4 \bmod 7$

$x=(a_1m_1y_1+a_2m_2y_2+ a_3m_3y_3) \bmod M$
$=((1*42*3)+(2*35*5)+(3*30*4)) \bmod 210$
$=836 \bmod 210$
$=206$

$42y_1=1 \bmod 5$
The value of y_1 lies in the interval $1 \le y_1 \le 5$
So, first substitute the value of y1 as 1…

$(42*1)\%5=2$

$(42*2)\%5=4$

$(42*3)\%5=1$

Therefore, the value of $y_1=3$

Verification
$(206\%5)=1$
$(206\%6)=2$
$(206\%7)=3$

Problem 2

A bag has contained number of pens if you take out 3 pens at a time 2 pens are left. If you take out 4 pens at a time 1 pen is left and if you take out 5 pens at a time 3 pens are left in the bag. What is the number of pens in the bag.

$x \equiv 2 \bmod 3$
$x \equiv 1 \bmod 4$
$x \equiv 3 \bmod 5$

$a_1=2$
$a_2=1$
$a_3=3$

$n_1=3$
$n_2=4$
$n_3=5$

$M=n_1n_2n_3$
$M=3*4*5=60$
$m_i=M/n_i$

m1=60/3=20
m2=60/4=15
m3=60/5=12

$m_iy_i=1 \bmod n_i$
$20y_1=1 \bmod 3$
$y_1=2 \bmod 3$

$15y_2= 1 \bmod 4$
$y_2=3 \bmod 4$

$12y_3=1 \bmod 5$
$y_3=3 \bmod 5$

$x=(a_1m_1y_1+a_2m_2y_2+ a_3m_3y_3) \bmod M$
$=((2*20*2)+(1*15*3)+(3*12*3)) \bmod 60$
$=233 \bmod 60$
$=53$

Verification

(53%3)=2 ,(53%4)=1, (53%5)=3

Problem 3

Find the integer that has a remainder of 3 when divided by 7 and 13. But it is divisible by 12.

$x \equiv 3 \bmod 7$
$x \equiv 3 \bmod 13$
$x \equiv 0 \bmod 12$
$a_1=3$
$a_2=3$
$a_3=0$

$n_1=7$
$n_2=13$
$n_3=12$

$M=n_1n_2n_3$

M=7*13*12=1092

$m_i=M/n_i$

m1=1092/7=156
m2=1092/13=84
m3=1092/12=91

$m_iy_i=1 \bmod n_i$
$156y_1=1 \bmod 7$
$y_1=4 \bmod 7=4$

$84y_2= 1 \bmod 13$
$y_2=11 \bmod 13=11$

$91y_3=1 \bmod 12$
$y_3=7 \bmod 12=7$

$x=(a_1m_1y_1+a_2m_2y_2+ a_3m_3y_3) \bmod M$
=((3*156*4)+(3*84*11)+(0*91*7)) mod 1092
=276

Verification

(276%7)=3

(276%13)=3

(276%12)=0

RSA

RSA is a best known and widely used public-key scheme by Rivest, Shamir and Adleman of MIT in 1977.

The **RSA** scheme is a cipher in which the plaintext and ciphertext are integers between 0 and *n* - 1 for some *n*. A typical size for *n* is 1024 bits. Encryption is denoted by E and decryption is denoted by D, plain text is denoted by M and ciphertext is denoted by D.

Public key: PU={e,n}
Private key: PR={d,n}

Both the sender and receiver must know 'n'. The sender knows 'e' amd the receiver knows 'd'.
The requirements to be satisfied by the algorithm are

(i) It is possible to find values of e,d and n such that $M^{ed} = M \bmod n$ for all M<n.
(ii) It is easy to calculate t M^e and C^d for all values of M<n
(iii) It is infeasible to determine d given e and n.

Key Generation

Select two prime numbers p and q, where $p \neq q$
Calculate n = p*q
Calculate $\varphi(n) = (p - 1)(q - 1)$
Select e such that e is relatively prime to $\varphi(n)$ and less than $\varphi(n)$.
Calculate d such that $de \equiv 1 \bmod f(n)$ and d<f(n). d is calculated using extended Euclid's algorithm.

Encryption

Plaintext: $M < n$

Ciphertext: $C = M^e \pmod n$

Decryption

Plaintext: $M = C^d \pmod n$

Example:

- Choose p = 3 and q = 11
- Compute n = p * q = 3 * 11 = 33
- Compute φ(n) = (p - 1) * (q - 1) = 2 * 10 = 20
- Choose e such that 1 < e < φ(n) and e and n are coprime. Let e = 7
- Compute a value for d such that (d * e) % φ(n) =
  1. One solution is d = 3 [(3 * 7) % 20 = 1]
- Public key is (e, n) => (7, 33)
- Private key is (d, n) => (3, 33)

The encryption $c = m^e \bmod n$

- $M = 2$
  $C = 2^7 \% 33 = 29$

The decryption $M = C^d \bmod n$

$C = 29$

$M = 29^3 \% 33 = 2$

KEY MANAGEMENT

There are two distinct aspects to the use of public-key cryptography:

I) The distribution of public keys

II) The use of public-key encryption to distribute secret keys

I) Distribution of Public Keys

There are four different schemes

i. Public announcement
ii. Publicly available directory
iii. Public-key authority
iv. Public-key certificates

(i) Public announcement

Any participant can send his or her public key to any other participant or broadcast the key to the community.

Limitation

Anyone can forge such a public announcement. That is, some user could pretend to be user A and send a public key to another participant or broadcast such a public key. Authentication is needed to avoid this problem.

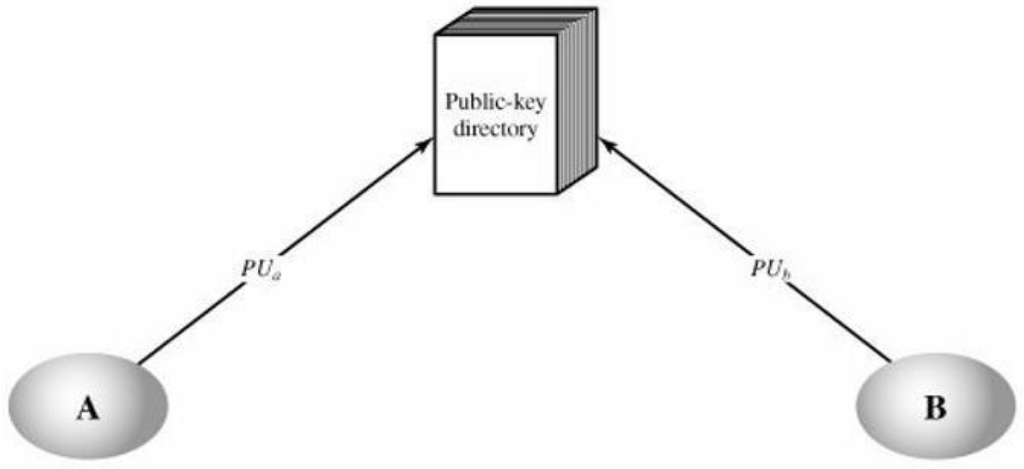
(ii) Publicly Available Directory

A greater degree of security can be achieved by maintaining a publicly available dynamic directory of public keys. Maintenance and distribution of the public directory would have to be the responsibility of some trusted entity or organization.

- The authority maintains a directory with a {name, public key} entry for each participant.
- Each participant registers a public key with the directory authority.
- Participants could also access the directory electronically.
- Participant may replace the existing key with new one at any time to avoid the attack on that key.
- Periodically, the authority publishes the entire directory or updates of the directory to all participants in the form of telephone index.

Advantage

More secure than individual public announcement.

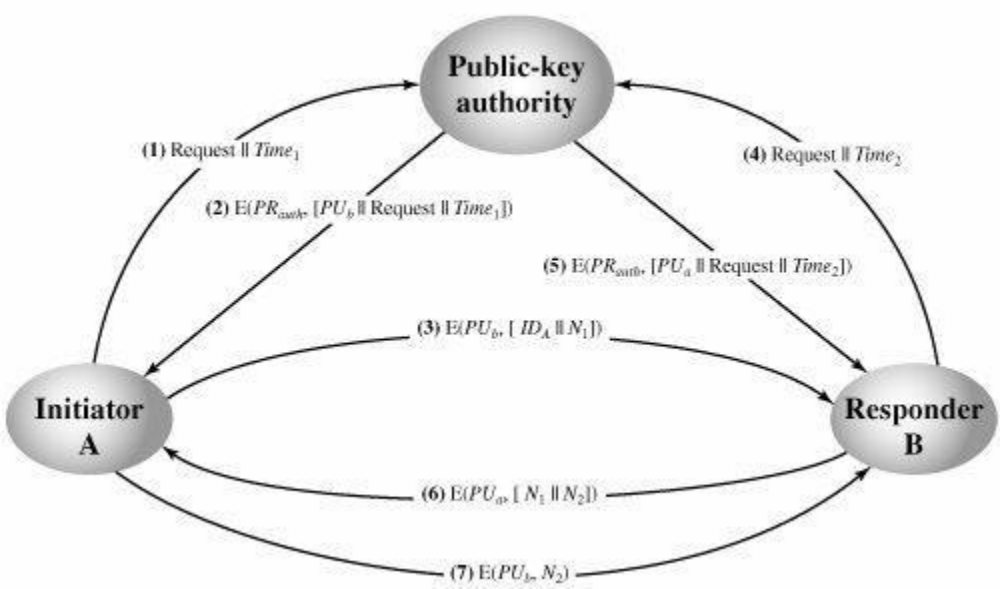


Limitation

Problem arises if the opponent captures the private key of the directory authority.

(iii) Public-key authority

Stronger security for public-key distribution can be achieved by providing tighter control over the distribution of public keys from the directory. The central authority maintains all public keys of participants in its dynamic directory. Each participant knows the public key of the authority, but the private key of the authority is kept secret.



1. A sends a timestamped request to public-key authority for the public key of B.
2. The authority replies with a message that is encrypted using the authority's private key, PRauth. A is know the public key of authority. Therefore, A decrypts the message. The message includes the following:
    - B's public key, PUb
    - Request already sent by A (for verification)
    - Time1 already sent by A (prove that the message is old or not)
3. A stores B's public key and send message to B in an encrypted format using B's public key. This message consists of
    - A's identity, (ID_A)
    - Nonce (N1), which is used to identify this transaction uniquely.
4. B retrieves A's public key from the authority in the same manner as A retrieved B's public key.
5. B sends a message to A encrypted with PUa and containing A's nonce (N1) as well as a new nonce generated by B (N2)
6. A returns N2, encrypted using B's public key, to assure B that its correspondent is A.

Advantage

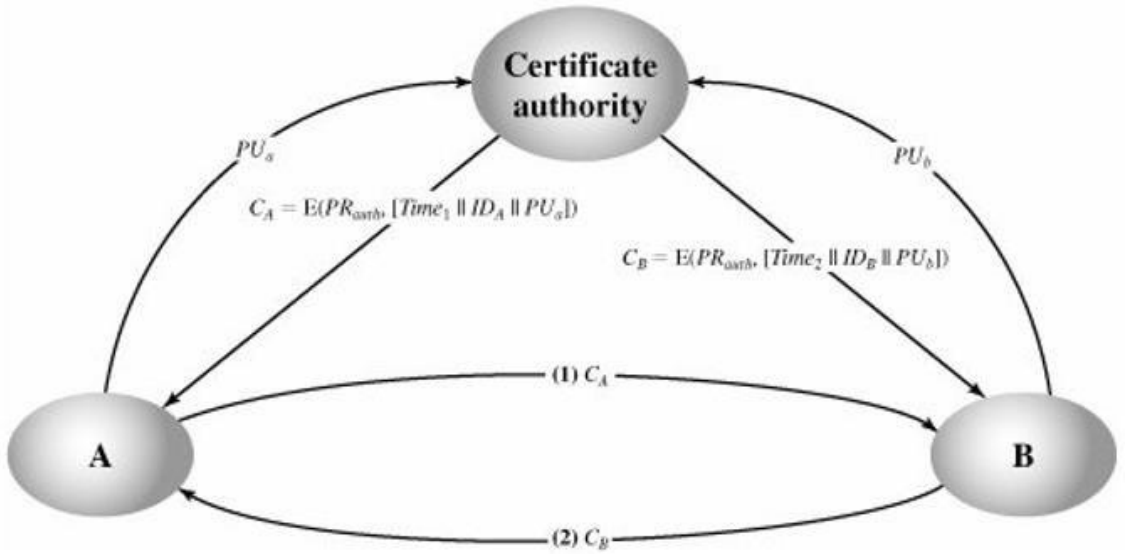
More secure and attractive than previous two.

Limitations

- Each and every time the user must appeal to the authority for a public key for every other user that it wishes to contact.
- The directory of names and public keys maintained by the authority is vulnerable to tampering.

(iv) Public key certificate

It uses certificates that can be used by participants to exchange keys without contacting a public key authority for its every transaction.



Each certificate contains a public key and other information created by certificate authority. Each participant conveys its key information to its corresspondant by transmitting their certificates. Other participant can verify that the certificate was created by the authority.

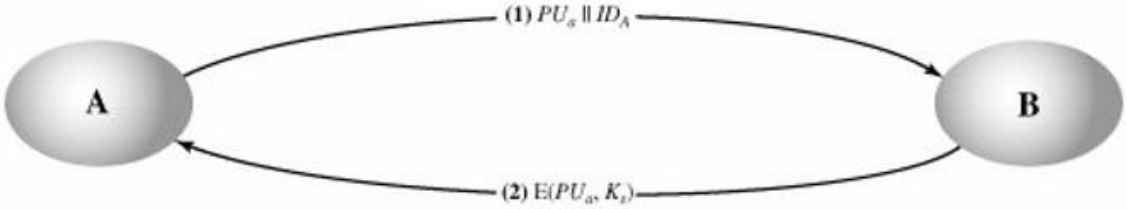
The requirements of the scheme are

1. Any participant can read a certificate to determine name and public key of the certificate owner.
2. Any participant can verify that the certificate originated from certificate authority.
3. Only the certificate authority can create and update the certificates.
4. Any participant can verify the currency of the certificate.

II) Public key encryption to distribute

(i) Simple secret key distribution
(ii) Secret key distribution with confidentiality and authntication

(i) Simple secret key distribution



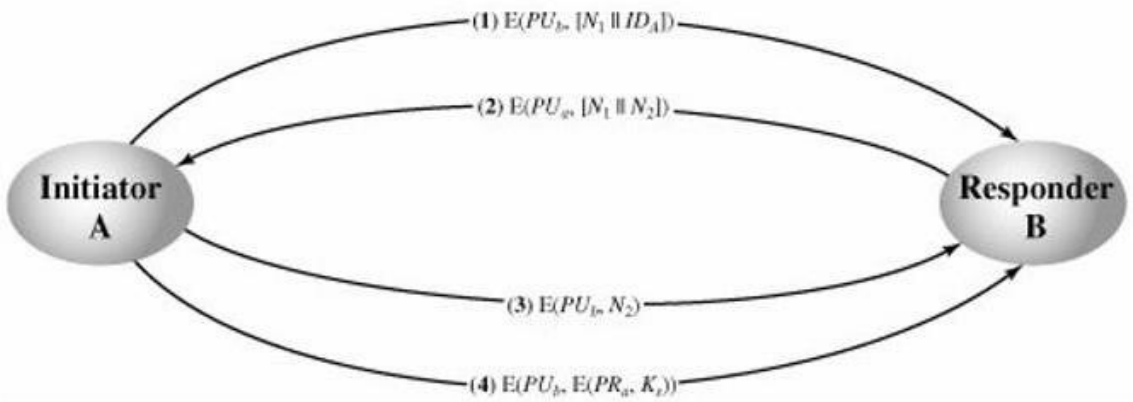
Steps used for communication between A and B are

1. A generates a public/private key pair {PUa, PRa} and transmits a message intended for B consisting of PUa and an identifier of A, IDA.
2. B generates a secret key, Ks, encrypted using A's public key and transmit to A.
3. A computes D(PRe, E(PUe, Ks)) to recover Ks. Now, both A and B knows Ks..

Once communication is over both a and B discard Ks.

Problem : Man in the middle attack . It can be rectified by using authentication.

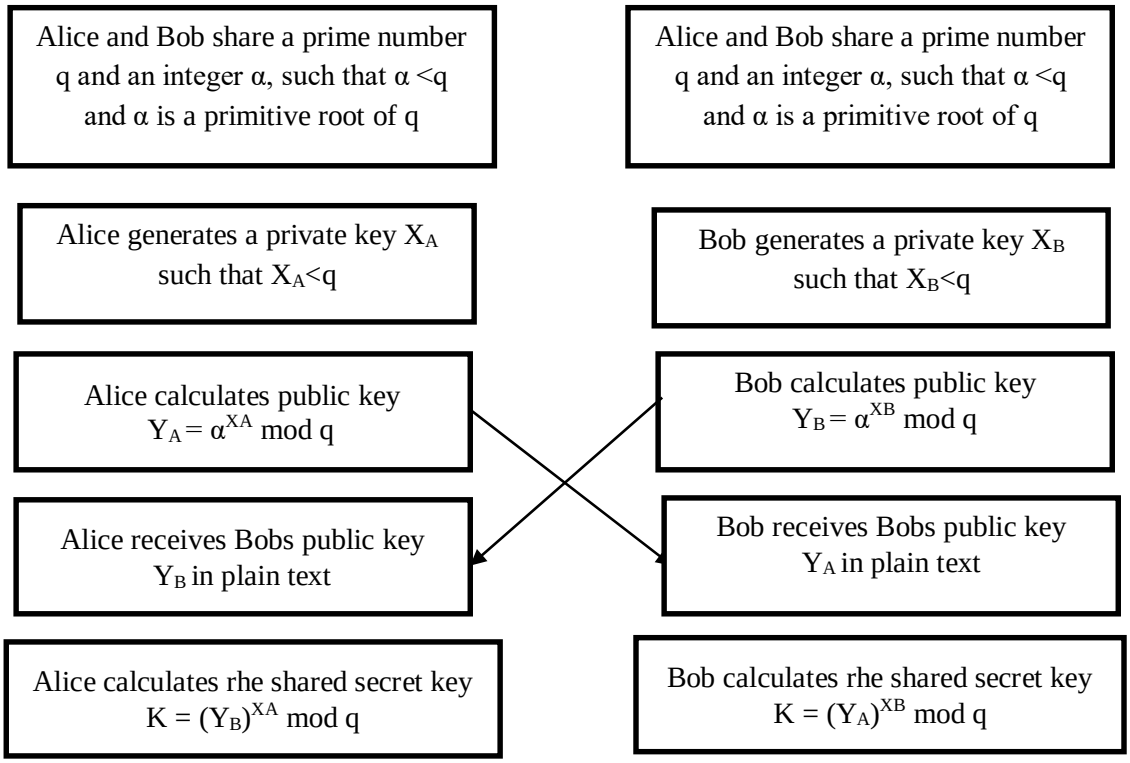
(ii) Secret Key Distribution with Confidentiality and Authentication



1. A uses B's public key to encrypt a message to B containing an identifier of A (ID_A) and a nonce (N1), which is used to identify this transaction uniquely.
2. B sends a message to A encrypted with PUa and containing A's nonce (N1) as well as a new nonce generated by B (N2).
3. A returns N2 encrypted using B's public key, to assure B that its correspondent is A.
4. A selects a secret key Ks and sends M = E(PUb, E(PRa, Ks)) to B. Encryption of this message with B's public key ensures that only B can read it; encryption with A's private key ensures that only A could have sent it.
5. B then computes D(PUa, D(PRb, M)) to recover the secret key.

DIFFIE – HELLMAN KEY EXCHANGE

The purpose of the algorithm is to enable two users to securely exchange a key that can then be used for subsequent symmetric encryption of messages. The algorithm itself is limited to the exchange of secret values.



For this scheme, there are two publicly known numbers: a prime number q and an integer that is a primitive root of q.

Suppose the users A and B wish to exchange a key, user A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{XA} \bmod q$.

Similarly, user B independently selects a random integer $X_B < q$ and computes $Y_B = \alpha^{XB} \bmod q$. Each side keeps the X value private and makes the Y value available publicly to the other side. User A computes the key as $K = (Y_B)^{XA} \bmod q$

User B computes the key as $K = (Y_A)^{XB} \bmod q$.

These two calculations produce identical results $K =$

$(Y_B)^{XA} \bmod q$

$= (\alpha^{XB} \bmod q)^{XA} \bmod q$

$= (\alpha^{\mathbf{XB}})^{XA} \bmod q$

$= \alpha^{XBXA} \bmod q$

$= (\alpha^{XA})^{XB} \bmod q$

$= (\alpha^{XA} \bmod q)$

$= (\alpha^{XA} \bmod q)^{XB} \bmod q$

$= (Y_A)^{XB} \bmod q$

Example:

Prime number, q=353

$\grave{\alpha} = 3$

A's private key, $X_A = 97$ B's private key, $X_B = 233$

A computes its public key, $Y_A = 3^{97} \bmod 353 = 40$. B computes its public key, $Y_B = 3^{233} \bmod 353 = 248$.

After they exchange public keys, each can compute the common secret key: A computes secret key, $K = (Y_B)^{XA} \bmod 353 = 248^{97} \bmod 353 = 160$.
B computes $K = (Y_A)^{XB} \bmod 353 = 40^{233} \bmod 353 = 160$.

Man-in-the-Middle Attack

The protocol is insecure against a man-in-the-middle attack. Suppose Alice and Bob wish to exchange keys, and Darth is the adversary. The attack proceeds as follows:

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2} and then computing the corresponding public keys Y_{D1} and Y_{D2}.
2. Alice transmits Y_A to Bob.
3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates $K_2=(Y_A)^{XD2} \bmod q$.
4. Bob receives Y_{D1} and calculates $K_1 = (Y_{D1})^{XB} \bmod q$.
5. Bob transmits X_A to Alice.
6. Darth intercepts X_A and transmits Y_{D2} to Alice. Darth calculates $K_1 = (Y_B)^{XD1} \bmod q$.
7. Alice receives Y_{D2} and calculates $K_2 = (Y_{D2})^{XA} \bmod q$.

At this point, Bob and Alice think that they share a secret key, but instead Bob and Darth share secret key K1 and Alice and Darth share secret key K_2.

All future communication between Bob and Alice is compromised in the following way:

1. Alice sends an encrypted message M: $E(K_2, M)$.
2. Darth intercepts the encrypted message and decrypts it, to recover M.
3. Darth sends Bob $E(K_1, M)$ or $E(K_1, M')$, where M' is any message. In the first case, Darth simply wants to eavesdrop on the communication without altering it. In the second case, Darth wants to modify the message going to Bob.

This vulnerability can be overcome with the use of digital signatures and public-key Certificates.

ELLIPTIC CURVE CRYPTOGRAPHY [ECC]

Elliptic curve cryptography [ECC] is a public-key cryptosystem. Every user has a public and a private key.

- Public key is used for encryption/signature verification.
- Private key is used for decryption/signature generation.

Elliptic curves are used as an extension to other current cryptosystems. **ECC- Algorithm**

Both parties agree to some publicly-known data items Therefore, the elliptic curve equation $y^2 = x^3 + ax + b \bmod p$ and values of a and b such that $4a^3 + 27\,b^2 \neq 0$

The elliptic group is computed from the elliptic curve equation A base point, G, taken from the elliptic group

Each user generates their public/private key pair

Private Key = an integer, x selected from the interval [1, p-1]

Public Key = product of private key and base point

(Product = x*G)

Example :

- Suppose Alice wants to send to Bob an encrypted message.
  - Both agree on a base point, G.
  - Alice and Bob create public/private keys.

Alice : Private Key = n_A

Public Key = $P_A = n_A * G$

Bob : Private Key = n_B

Public Key = $P_B = n_B * G$

Alice takes plaintext message, M, and encodes it onto a point, P_M, from the elliptic group.

Encryption : Alice choose another random k value from { 1,2,… p-1 } Cipher text : $C_m = \{ KG, P_m + KP_B \}$

Decryption : by Bob

Take the first point from Cm - KG

Multiply KG and private key of Bob : Product = $n_B KG$

Take the second point from Cm and subtract the product from it $P_m + KP_B - n_B KG$

Substitute $P_B = n_B * G$

Then $P_m + K\,n_B * G - n_B KG = P_m$

ECC is particularly beneficial for application where:

- computational power is limited (wireless devices, PC cards)
- integrated circuit space is limited (wireless devices, PC cards)
- high speed is required.
- intensive use of signing, verifying or authenticating is required.
- signed messages are required to be stored or transmitted (especially for short messages).

bandwidth is limited .

ELGAMAL CRYPTO SYSTEM

The ElGamal Cryptosystem is an entire public-key cryptosystem like RSA, but based on discrete logs

Let p large so secure and > m = message

Bob chooses prime p, primitive root g, Private key x

Compute public key y from x,p and g

Bob computes $y \equiv g^x \pmod p$

Bob publishes (p, g, y) and holds x secret

Bob chooses secret random number k, computes and Send C_1, C_2 to Alice the pair (C_1, C_2) where

Encryption Process:

$C_1 \equiv g^k \pmod p$

$C_2 \equiv y^k M \pmod p$ where M is Plaintext

Cipher text which sepeartes of two values (C_1, C_2)

Decryption Process

Alice calculates:

Bob chooses secret k, computes and sends to Alice the pair (C_1, C_2) where

- $C_1^x \equiv (g^k)^x \pmod p$
- $C_1^x \equiv (g^x)^k \pmod p$
- $C_1^x \equiv (y)^k \pmod p$
- Alice finds: $C_1^x \equiv (y)^k \pmod p$

  $C_2 / y^k \bmod p \equiv M \pmod p$

Example:

	BOB	**BOTH**	**ALICE**
KNOWS	Choose private key x=6 Compute $y=11^6 \bmod 23=9$	Let p=23 g=11	
CALCULATE		Public key y=9	
ENCIPHERS	Plain text M=10 Random key k=3 $C_1 \equiv g^k \pmod p = 11^3 \bmod 23=20$ $C_2 \equiv y^k M \pmod p = 9^3 * 10 \bmod 23=22$	$C_1=20$ $C_2=22$	
DECRYPTION KEY			$C_1^x \equiv (y)^k \pmod p$ $20^x \equiv (9)^3 \bmod 23$ $20^x \equiv 16 \bmod 23$
DECRYPTION MESSAGE			22/16=10mod23 M=10